Installing on Windows (manual)

If you should ever need to install RabbitMQ manually, this is how to do it:

Download the Server

Description	Download	
Packaged for Windows systems (zip)	**rabbitmq-server-windows-3.6.9.zip**	(Signature)

Install Erlang

Run the **Erlang Windows Binary File**. Erlang will appear in the Start Menu, and `\erlx.x.x\bin\erl.exe` will be in `C:\Program Files (x86)` or `C:\Program Files`, depending on your platform and whether you chose a 32bit or 64bit version of Erlang.

Set ERLANG_HOME

If you have an existing RabbitMQ installation with the broker running as a service and you installed an Erlang VM with a different architecture then you must uninstall the service before updating ERLANG_HOME.

Set ERLANG_HOME to where you actually put your Erlang installation, e.g. `C:\Program Files\erlx.x.x` (full path). The RabbitMQ batch files expect to execute `%ERLANG_HOME%\bin\erl.exe`.

Go to Start > Settings > Control Panel > System > Advanced > Environment Variables. Create the system environment variable ERLANG_HOME and set it to the full path of the directory which contains `bin\erl.exe`.

Install RabbitMQ Server

Download `rabbitmq-server-windows-3.6.9.zip` from the link above.

From the zip file, extract the folder named `rabbitmq_server-3.6.9` into `C:\Program Files\RabbitMQ` (or somewhere suitable for application files).

Finding Command Scripts and App Data

Commands for RabbitMQ Server

Within the `rabbitmq_server-3.6.9\sbin` directory are some scripts which run commands to control the RabbitMQ server.

The RabbitMQ server can be run as either an application or service (not both).

rabbitmq-server.bat starts the broker as an application.

rabbitmq-service.bat manages the service and starts the broker.

rabbitmqctl.bat manages a running broker.

Log in as an administrator. To see the output, run these from a **Command Prompt** in the `sbin` directory.

Note: On Windows Vista (and later) you will need to **elevate privilege** (e.g. right-click on the icon to select Run as Administrator).

Find Commands Easily

Set up the system path so you can find the server and `sbin` directory easily.

Create a system environment variable (e.g. RABBITMQ_SERVER) for "`C:\Program Files\RabbitMQ\rabbitmq_server-3.6.9`". Adjust this if you put `rabbitmq_server-3.6.9` elsewhere, or if you upgrade versions.

Append the literal string "`;%RABBITMQ_SERVER%\sbin`" to your system path (aka %PATH%).

Now you can run rabbitmq commands from any (administrator) Command Prompt.
You will need to navigate to `rabbitmq_server-3.6.9\sbin` to run commands if your system path does not contain the RabbitMQ `sbin` directory.

Synchronise Erlang Cookies (when running a manually installed Windows Service)

Erlang Security Cookies used by the service account and the user running `rabbitmqctl.bat` must be synchronised for `rabbitmqctl.bat` to function.

To ensure Erlang cookie files contain the same string, copy the `.erlang.cookie` file from the Windows directory (normally `C:\WINDOWS\.erlang.cookie`) to replace the user `.erlang.cookie`. The user cookie will be in the user's home directory (%HOMEDRIVE%%HOMEPATH%), e.g. `C:\Documents and Settings\%USERNAME%\.erlang.cookie` or `C:\Users\%USERNAME%\.erlang.cookie` (Windows Vista and later).

Finding Application Data

By default, the RabbitMQ logs and Mnesia database are stored in the current user's Application Data directory e.g. `C:\Documents and Settings\%USERNAME%\Application Data` or

`C:\Users\%USERNAME%\AppData\Roaming` (Windows Vista and later).

Execute `echo %APPDATA%` at a Command Prompt to find this directory. Alternatively, Start > Run `%APPDATA%` will open this folder.

Running RabbitMQ Server as an Application

The application is started by the `rabbitmq-server.bat` script in `sbin`.

Customise RabbitMQ Server Environment Variables

Environment variable defaults are set within the `rabbitmq-server.bat` file. You may need to **customise environment variables** for your installation.

Start the Broker as an Application

Run the command

```
rabbitmq-server -detached
```

Alternatively, you can double-click the `rabbitmq-server.bat` file in Windows Explorer.

If you start by double-clicking, a Command Prompt window opens, displays a short banner message, concluding with the message "completed with *[n]* plugins.", indicating that the RabbitMQ broker has been started successfully.

If you started without the `-detached` option, e.g. by double-clicking, you will need a second Command Prompt window to control the application cleanly. *Note:* Closing the original Command Prompt window will forcefully shut down a server started this way.

Stop or Manage the Broker

`rabbitmq-server` only starts the broker. To **manage the broker** use `rabbitmqctl` commands.

Running RabbitMQ Server as a Service

The service will run in the security context of the system account without the need for a user to be logged in on a console. This is normally more appropriate for production use. The server should not be run as a service and application simultaneously.

The service runs using the `rabbitmq-service.bat` script in `sbin`.

Customise RabbitMQ Service Environment Variables

Note: After setting environment variables, you may need to install the service again.

The `rabbitmq-service.bat` script recognises many of the same **environment variables** as `rabbitmq-server.bat`, as well as a few additional service environment variables.

Install the Service

Install the service by running

```
rabbitmq-service install
```

A service with the name defined by RABBITMQ_SERVICENAME should now appear in the Windows Services control panel (Start > Run `services.msc`).

Managing the Service

To manage the service (install, remove, start, stop, enable, disable), use **`rabbitmq-service.bat`** commands. You can also use the Windows Services panel (`services.msc`) to perform some of the same functions as the service script.

Start the Broker as a Service

To start the broker, execute

```
rabbitmq-service start
```

If the output from this command is "`Service RABBITMQ_SERVICENAME started`", then the service was started correctly.

Confirm the service named RABBITMQ_SERVICENAME reports a "Started" status in Services: Start > Run `services.msc`.

Port Access

Firewalls and other security tools may prevent RabbitMQ from binding to a port. When that happens, RabbitMQ will fail to start. Make sure the following ports can be opened:

4369: **epmd**, a peer discovery service used by RabbitMQ nodes and CLI tools

5672, 5671: used by AMQP 0-9-1 and 1.0 clients without and with TLS

25672: used by Erlang distribution for inter-node and CLI tools communication and is allocated from a dynamic range (limited to a single port by default, computed as AMQP port + 20000). See **networking guide** for details.

15672: **HTTP API** clients and **rabbitmqadmin** (only if the **management plugin** is enabled)

61613, 61614: **STOMP clients** without and with TLS (only if the **STOMP plugin** is enabled)

1883, 8883: (**MQTT clients** without and with TLS, if the **MQTT plugin** is enabled

15674: STOMP-over-WebSockets clients (only if the **Web STOMP plugin** is enabled)

15675: MQTT-over-WebSockets clients (only if the **Web MQTT plugin** is enabled)

It is possible to **configure RabbitMQ** to use different ports.

Default user access

The broker creates a user `guest` with password `guest`. Unconfigured clients will in general use these credentials. By default, these credentials can only be used when connecting to the broker as localhost so you will need to take action before connecting from any other machine.

See the documentation on **access control** for information on how to create more users, delete the `guest` user, or allow remote access to the `guest` user.

Managing the Broker

To stop the broker or check its status, use `rabbitmqctl.bat` in `sbin` (as an administrator).

Stopping the Broker
: Use `rabbitmqctl stop`.

Checking the Broker Status
: Use `rabbitmqctl status`. All rabbitmqctl commands will report the node absence if no broker is running (i.e. nodedown).

More **info on rabbitmqctl**

Logging

Output from the server is sent to a RABBITMQ_NODENAME.log file in the RABBITMQ_LOG_BASE directory. Additional log data is written to RABBITMQ_NODENAME-sasl.log.

The broker always appends to the log files, so a complete log history is retained.

You can rotate logs using `rabbitmqctl rotate_logs`.

Troubleshooting When Running as a Service

In the event that the Erlang VM crashes whilst RabbitMQ is running as a service, rather than writing the crash dump to the current directory (which doesn't make sense for a service) it is written to an `erl_crash.dump` file in the base directory of the RabbitMQ server (set by the RABBITMQ_BASE environment variable, defaulting to `%APPDATA%\%RABBITMQ_SERVICENAME%` - typically `%APPDATA%\RabbitMQ` otherwise).

Windows-specific Issues

We aim to make RabbitMQ a first-class citizen on Windows. However, sometimes there are circumstances beyond our control. Please consult the **Windows-specific Issues** page.

Getting Help

If you have questions or need help, feel free to ask on **RabbitMQ mailing list**.